Описание

Applepen - это большая торговая сеть, которая занимается продажей всего двух продуктов: яблок и карандашей. Ее магазины расположены в различных уголках Соединенных Штатов и более 10 лет обслуживают покупателей. Недавно топ-менеджмент компании решил более активно использовать имеющиеся у них данные в принятии решений. Каждый магазин собирает информацию о:

1. закупках (поставки яблок и карандашей два раза в месяц)
2. продажах (лог транзакций, по записи на каждую проданную позицию)
3. инвентарь (месячные данные общего количества яблок и карандашей на складе).

Данные доступны в формате CSV. Внутри файла данные отсортированы по дате.

Постановка задачи

Нам необходимо составить по этим данным три новые таблицы:

1. состояние склада на каждый день
2. месячные данные о количестве сворованного товара
3. агрегированные данные об объемах продаж и количестве сворованной продукции по штату и году.

Вклад

- Куракин Кирилл – написание Readme
- Насыров Тимур, Кун-Уота Романов – написание кода

Библиотеки

- ***Pandas*** — программная библиотека для обработки и анализа данных, построенная поверх библиотеки NumPy. В нашей задаче для облегчения записи Pandas будет обозначаться как pd
- ***NumPy*** — библиотека с открытым исходным кодом с такими возможностями, как поддержка многомерных массивов (включая матрицы), поддержка высокоуровневых математических функций, предназначенных для работы с многомерными массивами. В нашей задаче NumPy будет обозначаться как np.

Описание программы

Используя средства библиотеки Pandas: команду pd.read_csv('имя таблицы'), считаем данные из таблиц закупок, продаж и инвентаря в переменные формата DataFrame df_supply, df_sell, df_inventory соответственно и для наглядности выведем первые пять записей из каждого DataFrame-а командой DataFrame.head():

date	apple	pen
2006-01-01	35086	2730
2006-01-15	35002	2625
2006-02-01	34963	2759

date	apple	pen
2006-01-31	12157	811
2006-02-28	29859	2280
2006-03-31	42135	3317

date	sku_num
2006-01-01	MS-b1-ap-48914c5b-14d2-4b20-bdaf-b2ff5d9f4f0c
2006-01-01	MS-b1-ap-6baf7287-3e6a-4728-a3b1-8613de51eef8
2006-01-01	MS-b1-ap-83d7b005-c7d9-4deb-93a2-a8f7606d02b5

Внимательно посмотрим на логи, находящиеся в df_sell. Можно заметить, что в sku_num первые две буквы отвечают за название штата. Занесем название штата в переменную state.

Теперь создадим элемент DataFrame_res, в котором будут находиться данные по количеству товаров на складе на каждый день, распределенные по трем колонкам: дата, число яблок, число ручек. Для этого воспользуемся командой pd.DataFrame(columns = [‘date’, ‘apple’, ‘pen’])

Опять вернемся к структуре логов: в этот раз отметим, что 6-7 символы отвечают за тип транзакции – занося их в отдельную переменную, можно понять тип продаваемого продукта. Таким образом создадим цикл по дням, в котором мы будем:

- определять тип транзакции для каждого элемента логов (DataFrame.values – превращает текстовый элемент в NumPy для того чтобы можно было сравнивать элементы)
- запоминать текущую дату и агрегировать данные по продажам по дням. Определять находимся мы все еще в текущем дне или нет будем при помощи функции DataFrame.Shape – эта функция определяет размерность данных.
- заносить в новую таблицу cur_df продаж агрегированные данные и смещаться на один день вперед

Получив таким образом данные по ежедневным продажам, добавим их к данным по ежедневным поставкам: res_df = res_df.append(cur_df).

Функция DataFrame.append(DataFrame) складывает две переменные типа DataFrame по столбцам. Теперь для наглядности выведем первые пять элементов получившейся таблицы: res_df.head().

date	apple	pen
2006-01-01	33271	2574
2006-01-02	31409	2431
2006-01-03	29529	2260

Теперь создадим помесячную таблицу с количеством украденного товара. Нетрудно понять, что его можно подсчитать по простой рекурсивной формуле:

$$\text{Украденное} = \text{Товар на складе} - \text{Товар на бумаге} - \text{Уже подсчитанный украденный товар}$$

Товар на складе – это данные из уже полученной выше таблицы res_df, товар на бумаге – df_inventory. Подсчитанный украденный товар изначально равен нулю. При помощи небольшого трюка будем подсчитывать только по интересующему нас месяцу:

$$\text{real\_inv} = \text{res\_df}[\text{res\_df.date} == \text{x}[0]]$$

Занесем полученные помесячные данные по украденному товару в таблицу stolen_df и выведем ее:

date	apple	pen
2006-01-31	10	11
2006-02-28	6	6
2006-03-31	7	6

Наконец объединим две уже полученные таблицы в одну и соберем данные по годам. Для этого опять же создадим необычный цикл с условием:

$$if\ ind\ \%\ 24\ ==\ 23:$$
$$\text{cur\_date}\ =\ \text{cur\_date}[:\text{-}2] + \text{'31'}$$

Почему такое условие? В году 12 месяцев, а отчетность в начальных таблицах проводилась каждое начало и середину месяца, поэтому 12*2 = 24 раза за год проходила опись склада. Таким образом получаем на выходе для каждого магазина таблицу:

year	state	apple_sold	apple_stolen	pen_sold	pen_stolen
2006	MS	681341	78	52555	91
2007	MS	1362622	146	104494	167
2008	MS	2047732	224	156526	240